Marin Aranitasi, et. al. International Journal of Engineering Research and Applications
www.ijera.com
ISSN: 2248-9622, Vol. 12, Issue 4, (Series-IV) April 2022, pp. 54-57

RESEARCH ARTICLE OPEN ACCESS

Mobile malware detection techniques using system calls

Marin Aranitasi*, Alfred Daci**, Anisa Gjini ***
**(Department Basics of Informatics, Polytechnic University of Tirana, Albania*
*** (Department of Mathematics, Polytechnic University of Tirana, Albania*
****(Department Basics of Informatics, Polytechnic University of Tirana, Albania*

ABSTRACT
In the recent years the popularity of mobile devices has continuously increased. These devices are not used only by normal users to interact with each other or to navigate through the infinite world of internet and social media but also has become one of the most attractive tools for increasing the business profits. Along with this wide usage comes the risks. Android malware writers has targeted the mobile devices and specifically Android OS as the market leader. The detection of these malwares is one of the biggest issues concerning the scientific community.
This paper proposes a solution on detecting mobile malware in the Android OS by monitoring the anomalies of the system calls usage
Keywords **-** About five key words in alphabetical order, separated by comma

Date of Submission: 13-04-2022 Date of Acceptance: 29-04-2022

I. INTRODUCTION

Currently Android is the most user mobile operating system in the world. According to [1] android has over 2.8 billion active users and a market share of 75 %. Samsung is the largest manufacturer of Android smartphones.

Table 1. Android users

Year	Active users (in billion)
2012	0.5
2013	0.7
2014	1
2015	1.4
2016	1.7
2017	2
2018	2.3
2019	2.5
2020	2.8

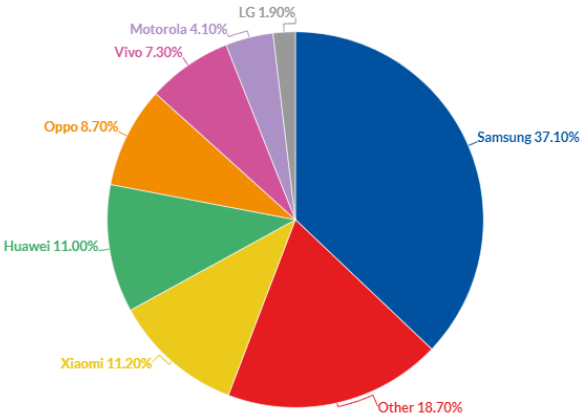

Fig.1 Android vendor market share

This massive use of the Android OS has increased the risks for the users. Android malwares has been growing exponentially. According to Kaspersky reports [2], there were 886.105 malicious installation packages in Q2 of 2021.

Fig.2 Number of detected malicious installation packages 2020-2021

Also, according to McAfee threat report [3] the new mobile malware detected in the third quarter of 2020 were 3.36 million. So, the situation on the security side is not that promising. For these reasons there are multiple efforts, from the academic and business world, in dealing with the increase of the security of mobile devices, and specifically for the Android OS. This paper presents a method for detecting the android malwares by detecting behavioral anomalies. As an implementation we use the system calls that are used by the Android applications. In this paper we monitor and analyze past behaviors of apps, in order to detect anomaly present ones. The rest of this paper is organized as follows. In section 2 we present the Related works on this specific topic. Section 3 is divided in 2 subsections and presents our solution for the security problems identified. The last section are the conclusions of this paper.

II. RELATED WORK

As mentioned earlier a lot of effort has been made to increase the security of mobile OS. Scientist has tried different ways to differentiate and analyze android malwares. One of the main issues that has been pointed out is the difficulty of the live malware detection on mobile devices. In this section we are going to present some of the most relevant and recent works on Android security and we will focus on those articles that mainly use system calls as their main feature to detect and then classify the malwares.

Chen Da et al. [4] proposed a new malware detection method that is based on system calls frequency. They do classifications of the applications and normalization to the system call usage in order to improve detection accuracy. Their method detected more than 93% of malwares.

Bathia et al. [5] developed a system that captures the system calls of all the applications while they are running on the device. Each interaction that is made triggers a system call that is collect by their system and then analyzed to classify the different behaviors of Android applications.

Hou et al. [6] has presented a dynamic analysis method that executes each routine of the Android apps as completely as possible. After extracting the system calls, they built the weighted graphs, that will be used by a deep learning framework in order to detect the android malwares. Their system, called Deep4MalDroid had also been integrated into a commercial antimalware software.

Vidal [7] has made a proposal for monitoring only the system calls during the boot process of the recent installed applications installed. This reduces the information that will be used in analysis. They proposed a system that uses pattern recognition with three processing layers: monitoring, analysis and decision making. They have made experiment on different datasets and presented their experimental results and proving that their approach is good when compared with analogous proposals.

III. HOW WE MIGHT APPROACH THESE ISSUES

In this paper we present a method that builds the normal behavior of the usage of the mobile device. We identify this normality by monitoring and analyzing the system calls of the applications. Normal behavior will be considered the "everyday" monitored behavior of the device and abnormality will be a "strange" behavior that was detected by the difference of the usage of the specific system calls that we decided to monitor. In the next section we are going to present our main feature of monitoring, the system calls, and after that the mathematical tool for analyzing and detecting the threats in the Android OS.

a. System calls

A system call is a routine that allows a user application to request actions that require special privileges. Adding system calls is one of several ways to extend the functions provided by the kernel. The distinction between a system call and an ordinary function call is only important in the kernel programming environment. User-mode application programs are not usually aware of this distinction.[8]. According to [9][10] three general methods exist for passing parameters to the OS as shown in figure 3:

1. Parameters can be passed in registers.
2. When there are more parameters than registers, parameters can be stored in a block and the block address can be passed as a parameter to a register.
3. Parameters can also be pushed on or popped off the stack by the operating system.

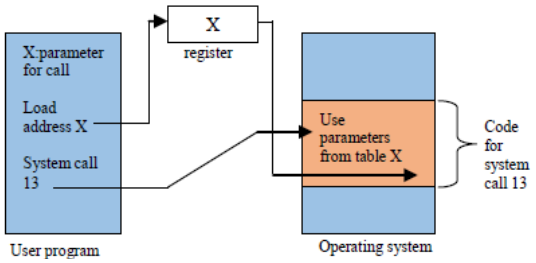

Fig. 3 System calls parameters

There are 5 different categories of system calls [9]:

1. **Process control** is a running program that needs to be able to stop execution either normally or abnormally. When execution is stopped abnormally typically, a dump of the memory is taken to be examined by a debugger.
2. The **file management** system calls include create(), delete(), read(), write(), reposition(), or close(). In addition, there is a need to determine the file attributes – get and set file attribute. Often the OS provides an API to make these system calls.
3. The **device management** process requires several resources to execute, if these resources are available, they will be granted, and control returned to the user process. These resources are also thought of as devices. Some are physical, such as a video card, and others are logical, such as a file. User programs *request* the device, and when finished they *release* the device. Like files, we can *read*, *write*, and *reposition* the device.
4. The **information management** system call exists for transferring information between the user program and the operating system. An example of this is *time*, or *date*. The OS also keeps information about all its processes and provides system calls to report this information.
5. The **communication** system call exists in two models of interprocess communication, the message-passing model and the shared memory model.

a. Message passing uses a common mailbox to pass messages between processes.

b. Shared memory use certain system calls to create and gain access to regions of memory owned by other processes. The two processes exchange information by reading and writing in the shared data

In this paper we are going to monitor 2 categories of system calls

1. Communication
2. File management

In the file management category, we have chosen to monitor and after analyse open (), read (), write (), close (). For communication category we have chosen we accept (), socket (), connect () system calls.

Table 2. System calls used

Types of system calls	System calls
Communication	accept (), socket (), connect ()
File management	open() read() write() close()

These are the system calls are chosen after careful study of the literature that states that these are the most common used by not genuine programs.

IV. MATHEMATICAL FOUNDATION OF THE PROPOSED SOLUTION

Mathematical modelling is a discipline that aims to transfer problems that arise within a certain scientific field or technological field to mathematical languages, so that the theoretical and numerical analyzes performed on these problems provide better information. Mathematical modeling is simply a representation of the problem in mathematical terms of a given idea. Essentially, they are characterized by making assumptions about variables (characteristics that change), parameters (characteristics that do not change), and functional relationships between variables and parameters that govern the dynamics of variables. Thus, mathematical models include hypotheses for the systems studied and allow us to compare these hypotheses with empirical data. Then it is necessary to express the working model in mathematical terms. To achieve this, we need to define the equations whose solutions describe it. The execution of this model allows us to perform simulations from which results and conclusions can be drawn. Interpreting such results and comparing them with the empirical data obtained from the observation of the real phenomenon will allow us to determine the efficiency of the mathematical model developed. If it is found that the predictions match what is happening in reality, we can assert that the model is appropriate, if not, it is necessary to start the modeling process again to get a more refined product. Most mathematical models are based on the use of ordinary differential equations or partial derivative differential equations because they are important pillars in Mathematical Modeling. Mathematical models developed to explore the behaviour of viruses (malwares) are based on models designed to study the behaviours of infectious diseases. This is because of the somehow

similarity between the behavior of biological viruses and those of malware (computer viruses). Undoubtedly, the pillars on which the models are based on differential equations are the Kermack [10] and McKendrick models or as it is known for short the SIR model which has the form:

$$\begin{cases} \frac{dS}{dt} = -\frac{\beta SI}{N} \\ \frac{dI}{dt} = \frac{\beta SI}{N} - \gamma I \\ \frac{dR}{dt} = \gamma I \end{cases}$$

Fig.4 SIRN model

where is the stock of susceptible population, is the stock of infected, is the stock of removed population (either by death or recovery), and is the sum of these three. The use of differential equations allows us to make a detailed mathematical analysis of the model in question. We can solve the model analytically or numerically using MATLAB or MAPLE software.. The case of malware viruses in mobile Smartphones and other mobile devices play a very important role in our lives seeing that there is a huge increase in recent years. Most of their applications require internet access, and consequently, they are exposed to the effects of malware. Consequently, as in the case of computer networks, predicting the behavior of mobile malware is very important. The model based on systems of differential equations is a mathematically coherent model, which provides a detailed study of the main characteristics of their dynamics: stability, equilibrium, etc. We will give more details in a second paper when we have the real data for the validation of the theory. S I R N

V. CONCLUSION

Mobile devices are widely spread and has become a very important part of our everyday life. As a result of this wide popularity many security problems have arisen. The diversity of such security problems is one of the issues that has triggered a widely response from the scientific community and from the private sector. Common ground has been found that each solution developed, after proper validation, has to be directly implemented in real life devices, otherwise the technological race between the "good" and "evil" will not have a happy end.

In this paper we assume that the Android OS that is under attack will behave abnormally. To sustain our assumption, we identify in the system calls the main feature that will help us find the abnormal behavior. Also, a mathematical method is presented in order to analyze the captured sys calls. As future work we plan to validate the method presented here, with real data. We also will refine our mathematical method in order to define more precisely the normality and abnormality.

REFERENCES

[1]. Business of Apps https://www.businessofapps.com/data/android-statistics/
[2]. Kasperksy reports https://securelist.com/it-threat-evolution-q2-2021-mobile-statistics/103636/
[3]. McAfee Lab Threats report https://www.mcafee.com/content/dam/global/infographics/McAfeeMobileThreatReport2021.pdf
[4]. Chen Da, Zhang Hongmei and Zhang Xiangli, "Detection of Android malware security on system calls," 2016 IEEE Advanced Information Management, Communicates, Electronic and Automation Control Conference (IMCEC), 2016, pp. 974-978, doi: 10.1109/IMCEC.2016.7867355.
[5]. T. Bhatia and R. Kaushal, "Malware detection in android based on dynamic analysis," 2017 International Conference on Cyber Security And Protection Of Digital Services (Cyber Security), 2017, pp. 1-6, doi: 10.1109/CyberSecPODS.2017.8074847.
[6]. S. Hou, A. Saas, L. Chen and Y. Ye, "Deep4MalDroid: A Deep Learning Framework for Android Malware Detection Based on Linux Kernel System Call Graphs," 2016 IEEE/WIC/ACM International Conference on Web Intelligence Workshops (WIW), 2016, pp. 104-111, doi: 10.1109/WIW.2016.040.
[7]. Vidal, J.M., Monge, M.A.S., Villalba, L.J.G.: A novel pattern recognition system for detecting Android malware by analyzing suspicious boot sequences. Knowl. Based Syst 150, 198–217 (2018)
[8]. IBM System calls https://www.ibm.com/docs/en/aix/7.2?topic=concepts-system-calls
[9]. Kansas State Polytechnic http://faculty.salina.kstate.edu/tim/ossg/Introduction/sys_calls.html
[10]. Riccardo Giubilei "*Closed form solution of the SIR model for the COVID-19 outbreak in Italy"* https://doi.org/10.1101/2020.06.06.20124313